

OpenCUDA+MPI

A Framework for Heterogeneous GP-GPU Cluster Computing

Kenny Ballou

May 3, 2013

- 1 Project Introduction
- 2 Project Goals / Objectives
- 3 Project Results
- 4 Future Work

1 Project Introduction

■ Project

2 Project Goals / Objectives

■ A Framework

■ Metrics of Goals

3 Project Results

■ Configuration

■ pyCUDA + MPI4PY

■ Initial Timing Results

4 Future Work

■ Develop Framework

■ Profiling and Testing

■ Cluster Configuration

Project

Parallel and Distributed Computing

What is GP-GPU Distributed Computing?

- Parallel:
 - Processing concurrently
 - CUDA
- Distributed
 - Processing over many computers, not necessarily in parallel
 - MPI
- Combined
 - CUDA+MPI

- 1 Project Introduction
 - Project
- 2 Project Goals / Objectives
 - A Framework
 - Metrics of Goals
- 3 Project Results
 - Configuration
 - pyCUDA + MPI4PY
 - Initial Timing Results
- 4 Future Work
 - Develop Framework
 - Profiling and Testing
 - Cluster Configuration

Framework

- Ease and abstract difficulties in Parallel / Distributed Programming
- Allow for Diversity in Computing Environment
 - “Jungle” Computing
- Add process “scheduler” to best utilize available computing resources
- Add Cluster Configuration and Management
- Release as FOSS

Goal Metrics

- Develop Several Test Programs
 - Vascular Extraction from CT angiography scans
 - N-Body Simulation (possible)
 - Prime Number Searching (possible)
- Profile / Compare CPU-only, CUDA-only, MPI+CUDA Solutions

- 1 Project Introduction
 - Project
- 2 Project Goals / Objectives
 - A Framework
 - Metrics of Goals
- 3 Project Results
 - Configuration
 - pyCUDA + MPI4PY
 - Initial Timing Results
- 4 Future Work
 - Develop Framework
 - Profiling and Testing
 - Cluster Configuration

Salt Node Configuration

- Provision Software/Settings/Daemons
- Bring up and down Nodes
- Complete for all nodes except “master”

pyCUDA + MPI4PY

- pyCUDA
 - Host to Device Memory Copies
 - Device to Host Memory Copies
 - Block, Thread Indexing Complexities
 - `gpuarray`
- MPI
 - `mpirun` is a bit messy
- MPI4PY:
 - No Python 3.x support

Timing Results / Comparison

Method	Time (s)	Total Time (s)
CPU Only	13.7	254.13
CUDA (Single Node)	13.83	4172
MPI + CUDA (7 nodes)	10.51	(average) 3177

Table : Computational Timing Comparison of 10^9 element wise vector summation

- **LOTS** of IO
- Bad Example

- 1 Project Introduction
 - Project
- 2 Project Goals / Objectives
 - A Framework
 - Metrics of Goals
- 3 Project Results
 - Configuration
 - pyCUDA + MPI4PY
 - Initial Timing Results
- 4 Future Work
 - Develop Framework
 - Profiling and Testing
 - Cluster Configuration

Begin Framework Development

- Create Abstraction For CUDA and MPI
 - Array Slicing Calculations
- Create custom MPI runner
 - Will Help with Scheduling

Add Profiling, Unit Testing, and Integration Testing

- Use existing Python and CUDA profiling tools
- Interface Profiling tools into Framework
- Tests for sense of “correctness”

Cluster Configuration / Management

- Add Salt Configuration for Master Node
- Research and Implement a distributed filesystem

OpenCUDA+MPI

A Framework for Heterogeneous GP-GPU Cluster Computing

Kenny Ballou

May 3, 2013